



How to use floating-point or MMX™ instructions in Ring 0 or a VxD under Windows® 95

Information for Developers and ISVs

From Intel® Developer Services
www.intel.com/IDS

March 1996

Information in this document is provided in connection with Intel® products. No license, express or implied, by estoppel or otherwise, to any intellectual property rights is granted by this document. Except as provided in Intel's Terms and Conditions of Sale for such products, Intel assumes no liability whatsoever, and Intel disclaims any express or implied warranty, relating to sale and/or use of Intel products including liability or warranties relating to fitness for a particular purpose, merchantability, or infringement of any patent, copyright or other intellectual property right. Intel products are not intended for use in medical, life saving, or life sustaining applications. Intel may make changes to specifications and product descriptions at any time, without notice.

Note: Please be aware that the software programming article below is posted as a public service and may no longer be supported by Intel.

Copyright © Intel Corporation 2004

* Other names and brands may be claimed as the property of others.

How to use floating-point or MMX™ instructions in Ring 0 or a VxD under Windows® 95

March 1996

CONTENTS

Implication

The Solution

How should you use floating-point or MMX™ instructions within your VxD

How to use floating-point or MMX™ instructions in Ring 0 or a VxD under Windows® 95

March 1996

The current release of Windows® 95 4.00 does not allow floating-point or MMX™ instructions within VxD's, which run in ring 0. Floating point and MMX™ instructions in applications and DLL's are not restricted. The reason for the restriction is because Windows® 95 does not allow floating-point exceptions when they are originated from ring 0.

Implication:

Systems using Intel processors with MMX™ technology, including OverDrive® processors with MMX™ technology will display a blue screen exception error message when attempting to run software which executes MMX™ instructions at the ring 0 level, under early versions of Windows® 95 (pre-OSR2 releases)

The solution:

The solution is incorporated in the version 4.02 of VMCPD VxD. The revised VxD is included in the new OEM version of Windows® 95 called OSR2 (see MS web page).

Distribution of the solution:

- OSR2 will be distributed on new machines after release.
- DirectX Beta 3 installs the new VMCPD VxD. Refer to MS documentation.

What should ISV do:

Within Windows® 95 OSR2, support for FP/MMX™ instructions in ring 0 is implemented in VMCPD.VxD version 4.02.

Ring 0 FPU/MMX instruction dependent software should detect the existence of VMCPD.VxD version 4.02, at installation time and at run time by checking the version number of the VMCPD. If an early VMCPD VxD version is detected, then Direct X applications should install the new version of Direct X (Beta 3 and above) to update the VMCPD VxD.

The following API will report the version of the VMCPD VxD:

`VxDCall VMCPD_Get_Version`

Version 4.02 or above will allow FP/MMX™ instruction usage within ring 0

How should you use floating-point or MMX™ instructions within your VxD?

There are two major cases where the VxD is called:

1. If the VxD is being called from an application using DevIOCcontrol then it runs on the same context of the calling thread. It uses the state and cleans it when it returns. The caller thread does not assume that the state is preserved.
2. If the VxD is being invoked by other means then:

```
CurrentThread = Get_Cur_Thread  
VMCPD_GET_THREAD (CurrentThread, MyVxD_Buff)  
MMX  
MMX  
...  
VMCPD_SET_THREAD (CurrentThread, MyVxD_Buff)  
RET
```